

Pizza Delivery Helper

Aldo Doronzo

2008 / 2009

Abstract

This is a report describing the Pizza Delivery Helper project realized during the course of Mobile Services taught by prof. Ricci at the Free University of Bolzano. The task was to build a complete mobile application able to run on real java-enabled mobile phones. Pizza Delivery Helper is an application, as suggested by its name, which aims to help a Pizza place to make fast and efficient deliverables to its customers. In particular the system architecture, the human computer interaction, the code structure and the major technical problems and related solutions are presented.

Contents

1	System Architecture	3
1.1	Server functionalities	3
1.2	Client functionalities	4
2	Human / Computer interaction	5
2.1	Server interaction	5
2.1.1	Setup wizard	5
2.1.2	Food menu management	5
2.1.3	Pizza boy management	5
2.1.4	Pizzeria profile management	5
2.1.5	Order management	9
2.2	Client interaction	9
2.2.1	Setup wizard	9
2.2.2	Order management	9
2.2.3	Server profile management	14
3	Code structure	14
3.1	Server Structure	14
3.1.1	Classes for the package orders	14
3.1.2	Classes for the package pizzaBoys	15
3.1.3	Class for the package start	16
3.1.4	Class for the package foodMenu	16
3.2	Client Structure	17
4	Major technical problems and related solutions	18
4.1	Planning phase	18
4.2	Graphical User Interface	18
4.3	Data Management	18
4.4	Compatibility problem	20
5	Conclusion	20

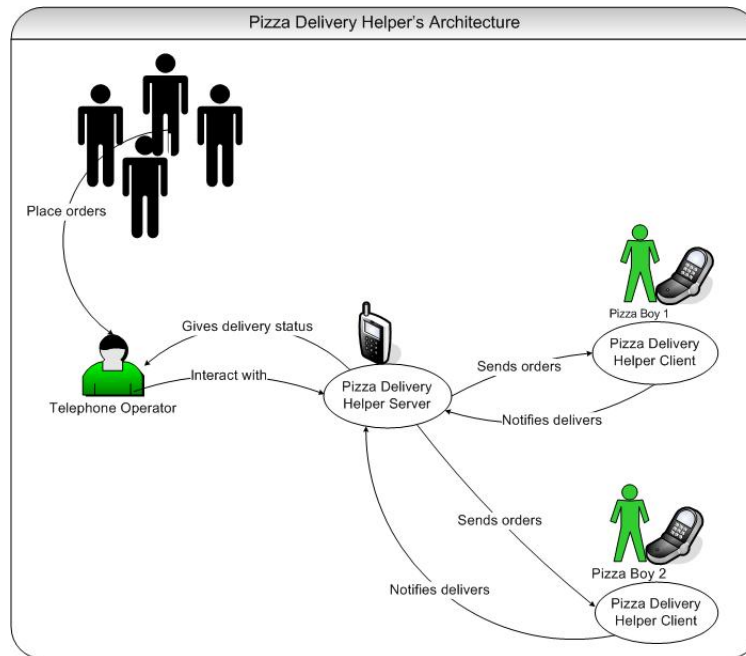


Figure 1: System Architecture

1 System Architecture

The Pizza Delivery Helper is a mobile application that helps a pizzeria to speed up and optimize pizza deliveries. It is perfect for businesses that have to manage many orders per hour and employ several pizza boys. This application is made by two components: a server and a client, both running on J2ME compliant devices. The components talk by exchanging common SMS messages. Finally, every data is persistently stored by means of J2ME record stores.

1.1 Server functionalities

How it is possible to observe from Figure 1, the Server component is installed on a mobile phone at the pizza place. This will be used by the telephone operator to input customers' orders and also to give customers good estimations about their deliverables' arrivals. The Server contains also a comfortable food menu that allows the Operator to input an incoming order with very few clicks. Finally, the server knows all the available pizza boys, so that the operator needs just to select one, and the server will take care to communicate the order to the right pizza boy. In particular, Server's functions are:

Pizzeria profile management. It is possible to specify the pizza place's profile and to edit it at any moment. The profile is made by business name, address and server number¹.

Food menu management. The food menu is divided in four categories: Pizza, Drink, Dessert, and Other. For each category, it is possible to add, edit, or delete items. Each item has: name, abbreviation and price.

Pizza boy management. It is possible to add, edit or delete boys in every moment. Each pizza boy has: name, surname, cell phone number, and vehicle description.

Order management. It is possible to add, or delete orders. An order is made by customer name, customer address, customer phone, and a list of ordered item. Also, every order as an assigned pizza boy and a total money due.

Order sending. It is possible to send orders as text messaging to the right pizza boy just with one click.

Delivered orders' acknowledgment receiving. When clients send acknowledgment of delivered orders, the server is able to recognize, identify and finally remove such orders from its order list.

Setup wizard. Last but not least, during server first run, a wizard helps the pizza place owner to correctly configure the Server on his phone going with him through the major configuration steps.

1.2 Client functionalities

The client is the software component that is installed on a pizza boy's mobile phone. Its functions are mainly two. The first is to receive and view orders sent by the Pizza place, and the second is to send to the Server acknowledgements about completed deliverables. In particular, Client's functions are:

Server profile management. It is possible to specify the server number and to edit it at any moment. This is extremely important, because if the number is absent or not valid, the client will not be able to communicate with the server.

Order receiving. Listening for incoming text messages allows the client to receive and store internally all the orders that a pizza boy needs to deliver.

Delivered orders' acknowledgement sending. When a pizza boy delivers an order and check it as done, the client will send an acknowledgement to the server.

¹It is the one of the cellphone on which the server component is installed.

Order viewing. In order to make a decent deliverable, a pizza boy needs to know exactly all the details of an order. For this reason, the client gives him all the information such as: customer name, address, cell phone number, etc.

Setup wizard. As in the server, also the client helps the pizza boy during client's first run going through the major configuration steps.

2 Human / Computer interaction

2.1 Server interaction

2.1.1 Setup wizard

When the Server runs for the first time, the wizard starts and takes the user through all the major configuration steps. You can find exactly all the wizard's windows in **Figure 2**.

2.1.2 Food menu management

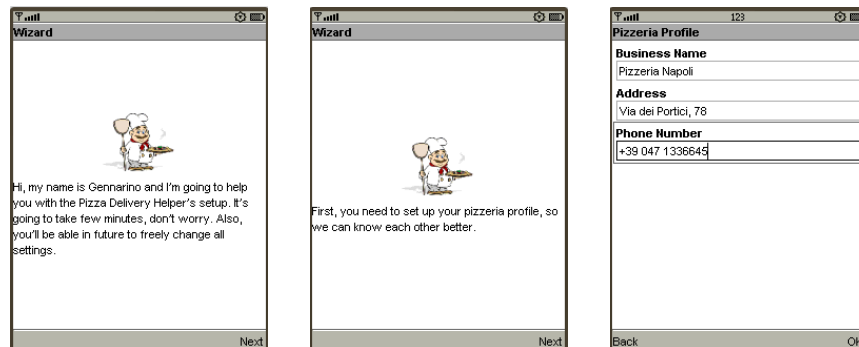
The Food menu is fundamental because it will determine the correctness of incoming orders. In **Figure 3**, it is possible to observe all the workflow that the user needs to go through, in order to obtain an effective food menu.

2.1.3 Pizza boy management

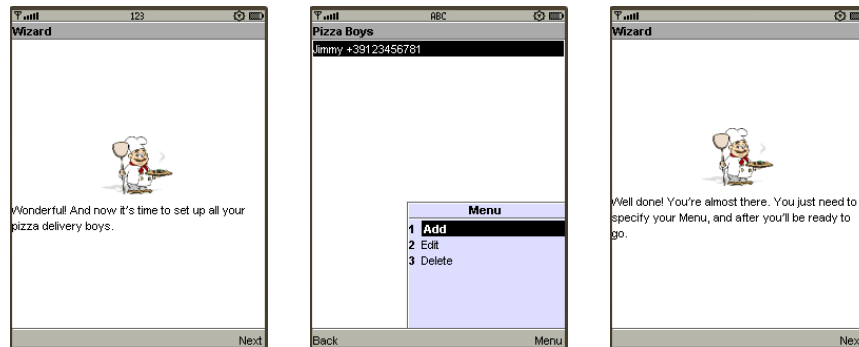
Pizza boys' setup is definitely a really sensitive setting. When an order is associated to a boy, the program sends automatically to the recorded mobile number of that boy. If that number is wrong then, the order will never make it to that pizza boy. For this reason, it is important to input data carefully. In **Figure 4**, it is possible to take a look at this process.

2.1.4 Pizzeria profile management

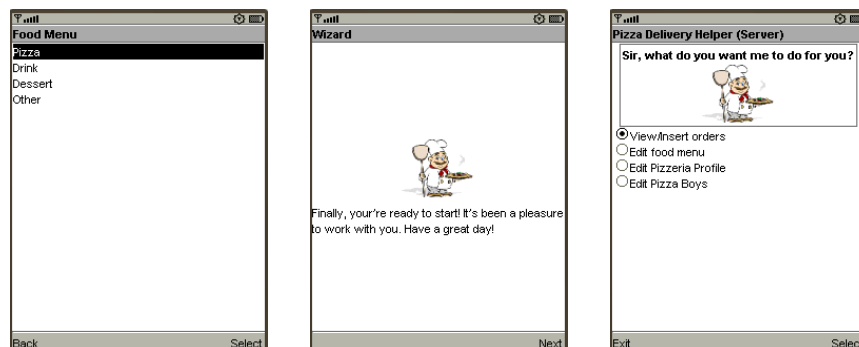
Despite the fact that a pizzeria profile should have been already defined during the wizard, it is always possible to change it. **Figure 5** shows it.



(a) The wizard helper, Gennarino, presents himself and explains the goal of this wizard. (b) Gennarino anticipates to the user the first setting that is the Pizzeria Profile. (c) The user can input pizzeria information.

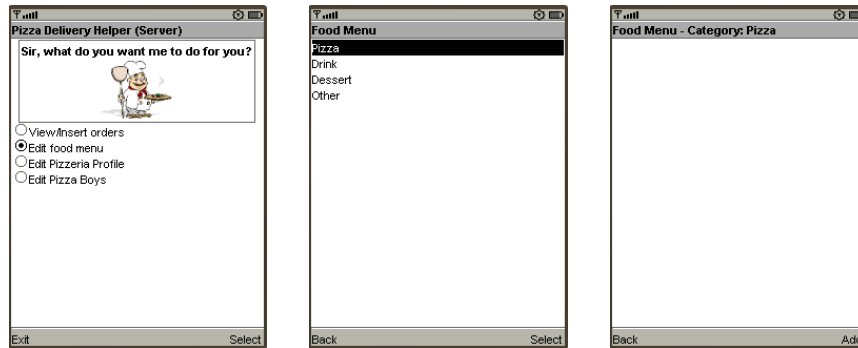


(d) Gennarino states to the user the second phase: pizza boys setting. (e) The user can Add, Edit and Delete pizza boys as he/she wishes. When the user has finished, he/she can go Back to the Wizard. (f) Gennarino announces the next phase: the food menu creation.



(g) The user can select the category of the food that he/she wants to inspect. When the user has finished, he/she can go Back to the Wizard. (h) Gennarino announces the end of the wizard and wishes the user all the best. (i) The wizard ends and the user sees for the first time the main menu of the application.

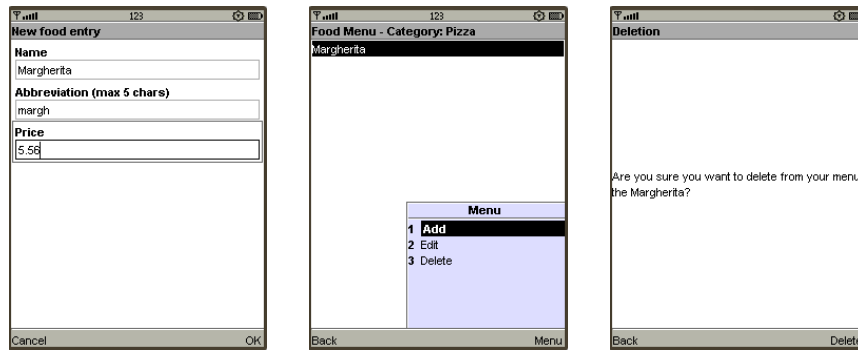
Figure 2: Setup wizard interaction



(a) In order to manage the food menu, the user selects Edit food menu from the Main menu.

(b) The user selects the category that wants to inspect.

(c) After a category has been chosen, an appropriate list is shown to the user.



(d) After the user clicks the Add button, a new window for a new food item entry appears. Here, the user can input name, abbreviation and price of the new food item.

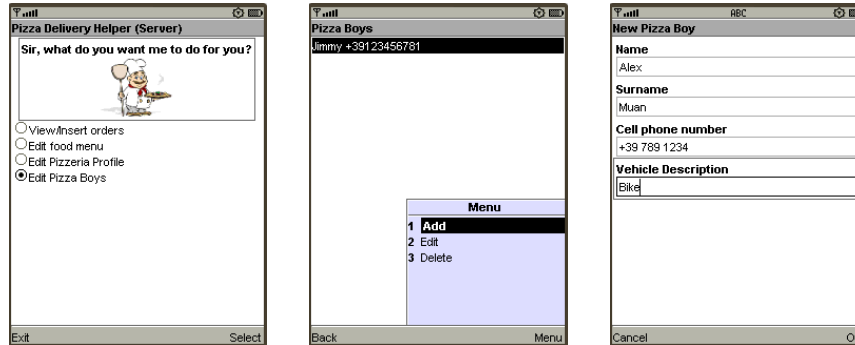
(e) Similar to c, the user can add, edit, or delete a food entry.

(f) When the user select a food entry and clicks on Delete, a confirmation window appears.

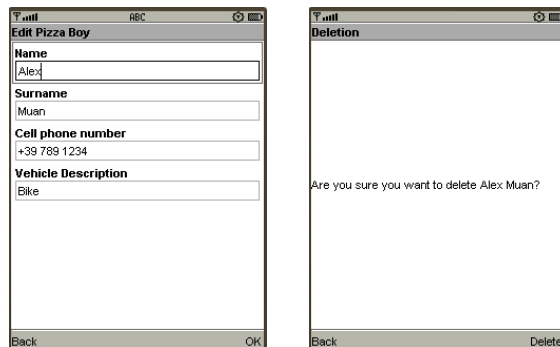


(g) When the user select a food entry and clicks on Edit, a window appears, to let him / her modify the item's details.

Figure 3: Food menu interaction

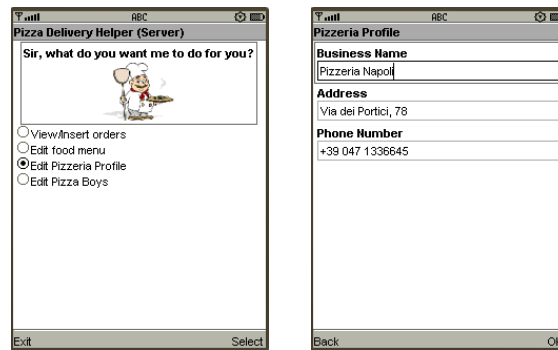


- (a) In order to manage the pizza boys, the user selects Edit Pizza Boys from the Main menu,
- (b) This is the list of all the known pizza boys. Here, the user can Add, Edit, or Delete one of them.
- (c) When the user clicks Add, a form for input a new pizza boy is shown to the user.



- (d) When the user selects an existing boy and clicks Edit, a form pre-filled form is shown. When the user clicks OK the changes are recorded.
- (e) When the user selects an existing boy and clicks Delete, a confirmation alert appear to make sure of user's intentions.

Figure 4: Pizza Boy interaction



(a) In order to change the pizzeria profile, the user selects Edit Pizzeria Profile from the Main menu. (b) A pre-filled form containing the pizzeria's info is presented to the user. When he / she clicks OK, changes are recorded.

Figure 5: Pizzeria profile interaction

2.1.5 Order management

This definitely represents the heart functionality of this application. When the telephone operator receives an incoming call from a customer, he / she will input the order as it shown in **Figure 6 and 7**.

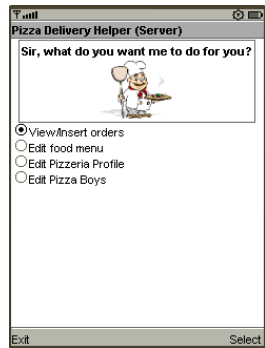
2.2 Client interaction

2.2.1 Setup wizard

When the Client runs for the first time, the wizard starts and takes the user through the configuration steps. You can find exactly all the wizard's windows in **Figure 8**.

2.2.2 Order management

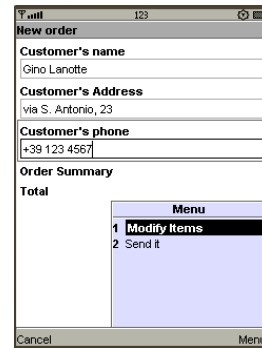
This is the most important part of the client and the definitely most used by the pizza boy. In **Figure 9**, it is possible to view all the human / computer interactions.



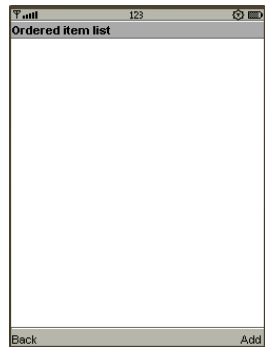
(a) When a user wants to manage orders, he / she needs to select View / Insert orders from the Main menu.



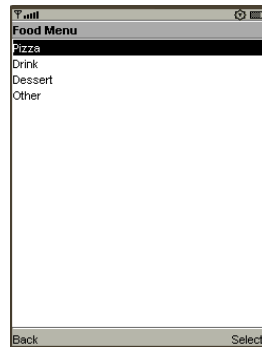
(b) When the order list is empty, then a user can only go back, or add a new order.



(c) After the user clicks Add, a new window is prompted and the user can input and see general order information. In order to add / delete an item to this order, Modify items must be clicked.



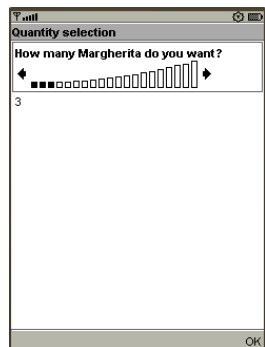
(d) This is the list of all the ordered items specific to this order. When it is empty, only Add is available.



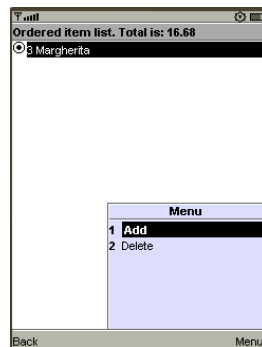
(e) When the user clicks Add, the food category menu is shown. The user can select in this way the category that he / she needs.



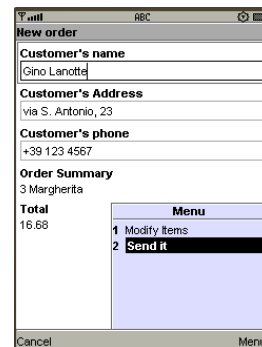
(f) When the user accesses to the category, he / she can select the item that he / she is interested in.



(g) After an item has been selected, the user selects the quantity.



(h) When also the quantity has been decided, the user returns to the list of the ordered item. Now, he / she can add or Delete items from the list.

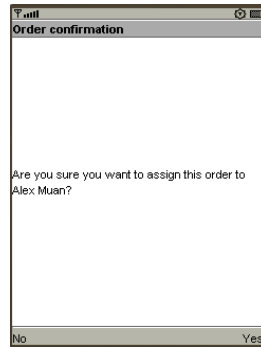


(i) When the user has finished with order input, he / she can send it.

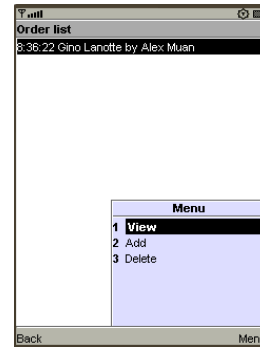
Figure 6: Order interaction



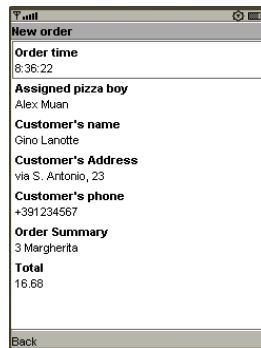
(a) When the user selects Send it, then he / she must choose who will have to deliver it.



(b) When a pizza boy has been chosen, then a confirmation alert is prompted to the user.

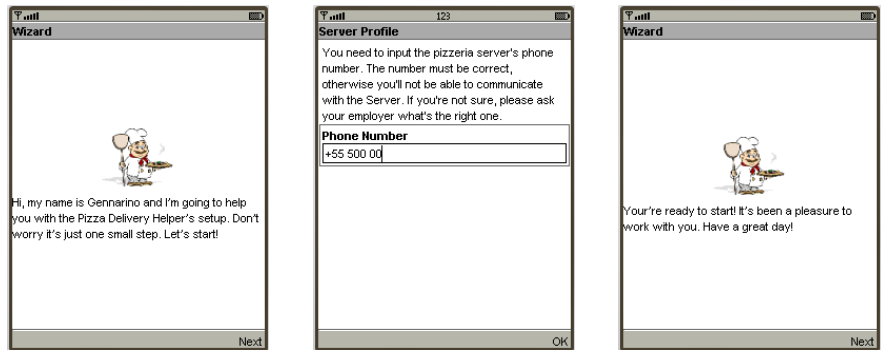


(c) When the user answers positively to the confirmation message, then the system will both send and locally record the order. After the user will see again the order list.

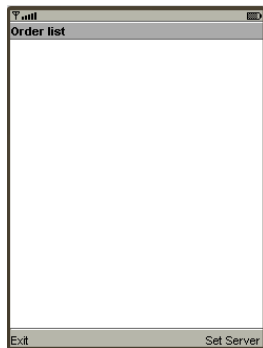


(d) When the user selects an order from the list and clicks on View, a summary window is prompted with detailed information order.

Figure 7: Order interaction

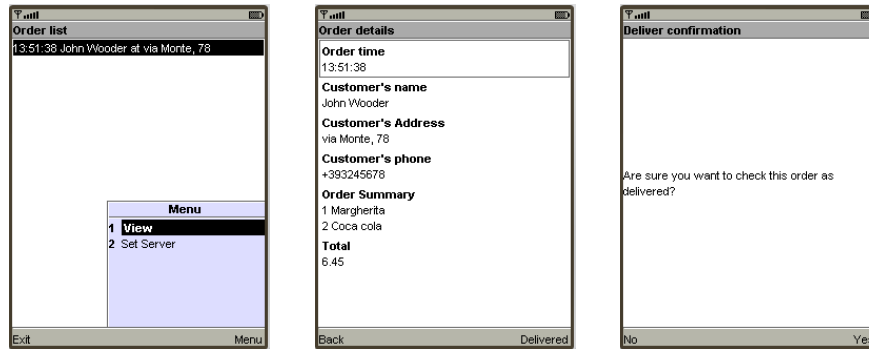


- (a) The wizard helper, Gennarino, presents himself and explains the goal of this wizard.
- (b) The user needs to input the right server's number.
- (c) Gennarino declares the end of the wizard.



- (d) When the wizard ends, the user will see the Main menu.

Figure 8: Setup wizard interaction

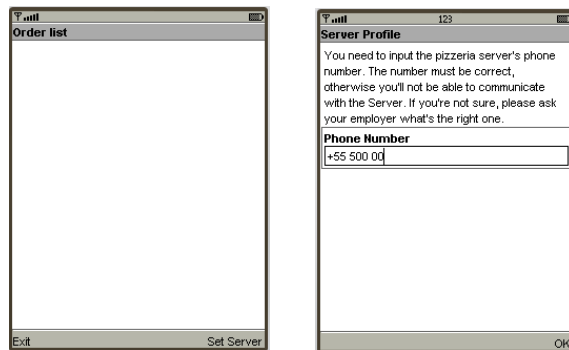


- (a) When an order will be received by the client, it will be shown in the Main menu.
- (b) When the pizza boy selects an order and clicks on View, then a summary window will prompt and give him / her all the order information.
- (c) When the pizza boy has delivered an order, he / she clicks on Delivered and then a confirmation message will be displayed.



- (d) If the pizza boy answer yes to the confirmation message, then an acknowledgment is sent to the Server, and the order is locally removed.

Figure 9: Orders interaction



(a) In order to change the Server profile, the user clicks on Set Server from the Main menu.
 (b) Here the user can change the Server's phone number.

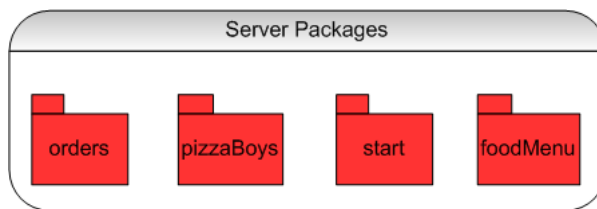
Figure 10: Server profile management

2.2.3 Server profile management

Despite the fact that a Server profile should have been already defined during the wizard, it is always possible to change it. **Figure 10** shows it.

3 Code structure

3.1 Server Structure



3.1.1 Classes for the package orders

Name	Description
Order	It represents the Order object. Each Order has time, summary, total, pizza boy name, customer name, customer address, and customer phone. In addition, this class supply a method to convert orders to a byte arrays.

OrderAdd	It is the Form seen by the user when he / she wants to add an order. Also, it responsible for input checking.
OrderDelete	It extends Alert and is the confirmation message that an user gets when he / she try to delete an order.
OrderList	It extends a List and is the main order window. Other than just listing orders, this class provides also the access to the order record store, and all the associated methods, to read, add and delete records. Finally, this class listens for incoming orders acknowledgments from clients. This task is done in a separated thread.
OrderPizzaBoyList	It extends List and is what the user sees when it selects a pizza boy for a specific order.
OrderSendConfirmation	It extends Alert and is the confirmation message that a user gets before sending a message.
OrderSender	It is the class responsible for sending Orders via SMS. It provides a public method called sendMsg that takes number, port and an order. However, the real sending is made in a separate thread.
OrderView	It is the Form seen by the user, when he / she wants to see the details of an order.
OrderedItem	It represents the object OrderedItem, that is made by a quantity and a food item.
OrderedItemDelete	It extends Alert and is the confirmation message a user gets before deleting an ordered item from an order.
OrderedItemList	It is the List of all the ordered item. This class uses a Vector as main data structure. Also, It provides to other classes methods to add and remove to / from the vector.
OrderedItemQuantity	It is the Form that helps the user to select a quantity of a specific ordered item.

3.1.2 Classes for the package pizzaBoys

Name	Description
------	-------------

PizzaBoy	It represents the object PizzaBoy. Each pizza boy has: name, surname, cellphone, and vehicle description. Also, it provides two important methods: loadPizzaBoy and savePizzaBoy. The former takes the name of the pizza boy record store, and returns an hash table containing all the pizza boys; the latter given an hash table and a name of a record store, it saves the hash table into the record store. Finally, this class offers methods for encoding a PizzaBoy in a byte array, and viceversa.
PizzaBoyAdd	It extends Form and is the one a user gets when he adds a new pizza boy.
PizzaBoyDelete	It is a confirmation Alert that the user gets when he / she tries to delete a pizza boy.
PizzaBoyEdit	It extends Form and is the one a user gets when he edits a pizza boy.
PizzaBoyList	It extends List and is the list of all the pizza boys known to the system. It offers a method to keep updated the list.

3.1.3 Class for the package start

Name	Description
PizzeriaProfile	It represents the Form that allows to input information regarding the pizzeria profile. Also, it provides methods for saving and reading such information to / from a record store. Finally, it gives method to encoding a pizzeria profile into a byte array, and viceversa.
ServerGUI	This class is the Midlet and is also the one providing the main menu.
WizardFirstScreen	This is the class that manages all the wizard work flow.

3.1.4 Class for the package foodMenu

Name	Description
FoodItem	It represents the FoodItem menu. Each food menu has: category, name, abbreviation, and price.

FoodItemAdd	It extends Form. It is the window a user uses to add a new FoodItem.
FoodItemDelete	It is the confirmation Alert a user sees when he / she tries to delete a food item.
FoddItemEdit	It extends Form. It is the window a user uses to edit a food item.
FoodItemList	It is the List that contains all the food items. Also, it provides methods for add, update, and delete food items from the record store.
FoodMenu	It is the List of food category.

3.2 Client Structure

Class Name	Description
ClientGUI	It is the Midlet and it also provides the client's main menu. Also, with an other thread, this class is able to receive sms containing orders to be delivered by the pizza boy. Other than viewing the list orders, this class provides methods to read, add and remove order records from the record store.
Order	It represents the Order object. Each Order has time, summary, total, pizza boy name, customer name, customer address, and customer phone. In addition, this class supply a method to convert orders to a byte arrays.
OrderDoneSender	It is the class responsible for sending done order acknowledgments via SMS to the Server. It provides a public method called sendMsg that takes number, port and an order. However, the real sending is made in a separate thread.
OrderView	It is the Form seen by the user, when he / she wants to see the details of an order. Also, from here the pizza boy can check an order as delivered.
ServerProfile	This is the class responsible for managing the server profile. By extending Form, it is able to show the Server profile to the user. Also, it provides methods to read, and save server information from / to a record store.
WizardFirstScreen	This is the class that manages all the wizard work flow.

4 Major technical problems and related solutions

4.1 Planning phase

Planning phase has been definitely my first issue to resolve. The problem was clearly stated in my mind, and I understood the domain of my application. In addition, I had also written the specifications in plain English. However, I was still missing details, and I soon realized that something was still missing. So I talked with my professors about it, and they helped me understand what I was missing. They explained me that in this kind of project is essential to use the interaction design. And so I did.

Using Microsoft Visio, I drew all the human / computer interactions of both Server and Client components. In Figure 11, it is possible to observe part of the output of my server's interface design. This really helped me, and finally made me focus more on the project's technical aspect: programming.

4.2 Graphical User Interface

Despite the fact that I had the interface's logic on paper, realizing it in J2ME was definitely another problem. Also, being my first project with a serious Java GUI interface did not make things easier. I thought the *top down* strategy would helped me.

I decided to start from the Server because of its importance in my system architecture. In addition, I divided the Server into functional areas, and I obtained in this way four packages. At his point, for each package I began to represent each window of my interface design into a java class. This approach helped me in keeping the logic easy, since each object had very few commands.

Certainly not having a class that globally manages all the GUI, made me passing reference each time an object calls another one. However, this gave me an extremely useful advantage, the window independence in going back and forward. For example, many of my classes can be called by any other classes. This means that changing the logic of my program, as well as reusing my code, are very easy things.

I took this advantage during the wizard. In fact, when the wizard calls another object, it passes its reference to it, so that called object knows exactly how to go back, when the user wants it. Also, I can call the same object from the main menu just passing the caller's reference, so that the called one, again, knows how to go easily back.

4.3 Data Management

Data management represented for me another challenge. At the beginning, I mistakenly thought that record stores where good only at storing data, but not really at managing data. As a consequence, I did the first data management

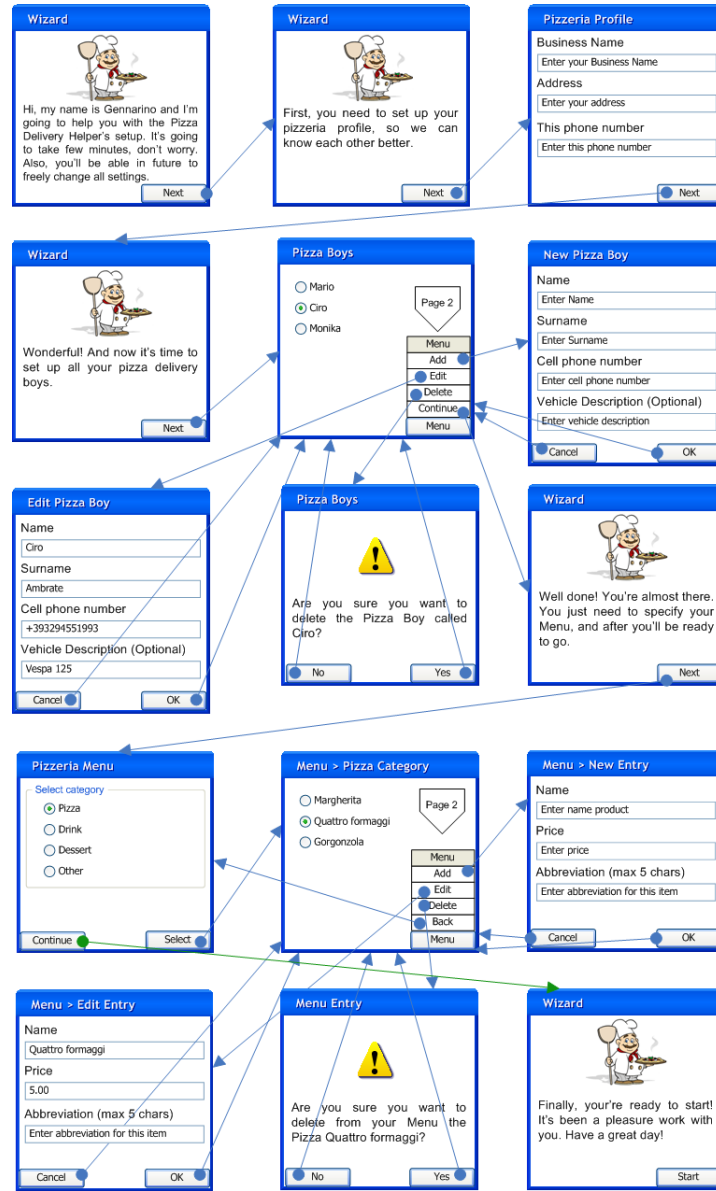


Figure 11: Part of my server's interface design

of my first part, the PizzaBoy ones, with a strategy not really performance optimizing. For example, when PizzaBoy is created, it loads all the datastore contents into an array, this data are shown to the user that makes some changes, after the datastore is made empty, and finally, all data are again saved.

Further readings let me to decide for a different and definitely better strategy. Apart from PizzaBoy and Pizzeria Profile that were my first classes, the others made changes on the fly, without having to load and save the datastore each time. This has been possible using the comfortable RecordFilter and RecordComparator interfaces.

4.4 Compatibility problem

Programming with J2ME made me understand that this ubiquitous platform suffers sometimes of some compatibility problems. It is not really a Sun's problem, rather more a single manufacturers' implementations one's. In fact, sometimes a J2ME is not always available and it has different behavior depending on which phone is running. For example, I experienced problems with the PlatformRequest for making external calls. Reading on the web let me change my mind about the implementation of this function in my application, since it seems to have strange behavior, especially on old phones. In fact, on old Nokia phones, when a Midlet requests a call, the phone closes this Midlet and makes the call, but when the call finishes, it does not return to the caller application which requested this call.

Other compatibility problem was the icon size of a Midlet. As several web sites state:

Many sellers, especially the operators, insist on an application having a build with the correct icon size in it for every phone that can show icons. This is a serious cause of resource fragmentation but an operator test lab WILL turn down a build just because it has an incorrect icon size.[2]

5 Conclusion

This course, and especially the project, taught me many things and opened my eyes on a reality that otherwise it would have remained hidden to me. About the application, I am quite happy with the output result, and I think that would be extremely easy in the future to develop other plugins for it. For example, it would be useful to develop a customer client version, where a customer is able to place orders directly to the Pizzeria just with his / her phone. It is absolutely easy to do, since it would be just a slighted modified version of the actual Server. I expect to develop it right after my summer exam session.

References

- [1] Mobile Services handouts, prof. Francesco Ricci,
<http://www.inf.unibz.it/~ricci/MS/index.html>
- [2] J2me forums.com, http://www.j2meforums.com/wiki/index.php/Application_Icon_sizes